To: J3
From: Dick Hendrickson/JOR
Subject: Renaming Defined Operators
Date: 5/28/98
Reference: WG5 #41 / **M15**

Specs:
Currently the USE statement allows things with names to be renamed. But, defined operators don't have *names* (in the bnf sense) and therefore can't be renamed. The reasons for allowing renaming of named things also apply to operator "names"; avoiding clashes with other local "names" and choosing more mnemonic "names". Part of the problem is that "name" is defined to be an alphanumeric string and a "defined operator" is either ".letters." or an extended intrinsic operator. This proposal only allows renaming of the defined-unary-op and defined-binary-op operators, not the extended-intrinsic-op class. There are two many difficulties with trying to rename .NOT., * or = to make it useful.

Defined operators can be in the *only-list*. This proposal extends the *only-list* in the natural way to allow operators to also be renamed in the *only-list* and then the syntax is also added to the *rename-list*. We don't have to worry about genericness of the renamed operators. The current rules for resolving overloads will work the same way as they do for unrenamed operators.

Syntax:
Currently a USE statement with an ONLY can specify an operator and rename a name as in:

```
USE  your_module, ONLY : OPERATOR(.YourOperator.), my_name => your_name
```

The proposal is to allow renaming of OPERATORs with

```
OPERATOR (.MyOperator.) => OPERATOR(.YourOperator.)
```

The rename list on an USE statement without an ONLY has the same form as the rename part of an ONLY list and has the same extension

```
USE your_module, OPERATOR(.MyOperator.) => OPERATOR (.YourOperator.)
```

For both cases, there will be a constraint requiring .YourOperator. to be a defined unary or binary operator in the module and requiring .MyOperator. to be a legal defined unary or binary operator in the local scope and not be the same as an extended intrinsic name (*, .NOT., etc.).

Outline of Edits (to 007R1):

The easy edits are
P211, add OPERATOR to R1110 so it becomes
R1110 *rename* **is** *local-name => use-name*
or OPERATOR(*local-defined-operator*) => OPERATOR(*use-defined-operator*)

add OPERATOR to R1113 so it becomes
R1113 *only-rename* **is** *local-name => use-name*
or OPERATOR(*local-defined-operator*) => OPERATOR(*use-defined-operator*)

Constraint: Each *use-defined-operator* shall be a public *defined-binary-op* or *defined-unary-op* in the module.

Pseudo-constraint: *local-defined-operator* must be a *defined-binary-op* or *defined-unary-op* and not the same as an existing intrinsic operator.

The hard edits are to the text in 11.3.2. Currently for renamed things it says the "local name" is either the "local name" from the rename clauses or the "real name" from the module or both, depending on how many different USE statements there are for the module. We either need to add parallel text describing how operators have a "local "name"", or say something like "the local defined operator name" behaves just like the "local name" behaves, or come up with a new term, "local things", which is either a "local name" or "local defined operator". We also need changes to 14.1.2.3 to make sure that it is the "local operator name" that is generically resolved. And we need to do it in a way that doesn't mess up the way *operator-names* can be used in their own module.